RTI Queuing Service

Release Notes

Version 5.2.0

Your systems. Working as one.

Printed in U.S.A. First printing.
June 2015.

Technical Support
Real-Time Innovations, Inc.
232 East Java Drive
Sunnyvale, CA 94089
Phone: (408) 990-7444
Email: support@rti.com
Website: https://support.rti.com/

Contents

Release Notes

1 Supported Platforms

RTI® Queuing Service is supported on the platforms in Table 1.1.

Table 1.1 **Supported Platforms**

Platform	Operating System	RTI Architecture
Linux	CentOS 6.0, 6.2 - 6.4	i86Linux2.6gcc4.4.5 x64Linux2.6gcc4.4.5
	Red Hat® Enterprise Linux 5.0	i86Linux2.6gcc4.1.1 x64Linux2.6gcc4.1.1
	Red Hat Enterprise Linux 6.0 - 6.5	i86Linux2.6gcc4.4.5 x64Linux2.6gcc4.4.5
	Red Hat Enterprise Linux 7	i86Linux3.xgcc4.8.2 x64Linux3.xgcc4.8.2
	Ubuntu Server 14 (3.x kernel)	i86Linux3.xgcc4.8.2 x64Linux3.xgcc4.8.2
Mac OS X	All OS X platforms listed in the *RTI Connext DDS Core Libraries Platform Notes* for the same version number	
Windows®	All Windows platforms listed in the *RTI Connext DDS Core Libraries Platform Notes* for the same version number	

2 Compatibility

Queuing Service is built on top of, and intended for use with, *RTI Connext™ DDS* with the same version number.

3 What's New in 5.2.0

3.1 Hello World and Remote Administration Examples

This release provides Hello World and Remote Administration examples for C++ and .NET.

The .NET **hello_world** example uses the *Queuing Service* wrapper API. The C++ **hello_world** example uses *DataWriters* and *DataReaders* directly to interact with *Queuing Service,* since the wrapper API is not available for C++.

The examples are in **<*your home directory*>/rti_workspace/5.2.0/examples/queuing_service**. (See also: Paths Mentioned in Documentation, in the *Queuing Service Getting Started Guide*.) For more information, see the **README.txt** files provided with the examples.

3.2 Reduced Memory Footprint of Empty SharedReaderQueues

In previous releases, the memory footprint of an empty SharedReaderQueue was 4MB. In this release, the memory footprint has been reduced to less than 1MB.

4 What's Fixed in 5.2.0

4.1 Replication Enabled at Service Level did not Work in Some Cases with SharedReaderQueues Created Using Remote Administration

Enabling SharedReaderQueue replication at the service level using the tag <service_qos>/<shared_reader_queue_replication> did not work in some cases. It did not work for SharedReaderQueues that did not explicitly enable replication using the tag <queue_qos>/<replication> and that were created at run-time using remote administration.

As a result, assuming there were two replicas:

- A Consumer would have obtained twice the number of messages, since replication was not enabled.
- Samples would have been positively acknowledged by *Queuing Service* even when instance quorum was not reached.

Specifically, enabling replication at the service level did not work when:

- <service_qos>/<shared_reader_queue_replication>/<enabled> was not explicitly set to true

or

- <service_qos>/<shared_reader_queue_replication>/<replication_settings> was not present

For example, the following configuration did not work:

```
<queuing_service name="RemoteConfigQS">
    <replication_settings>
       <queue_instances>3</queue_instances>
       <synchronize_consumer_assignment>
          false
       </synchronize_consumer_assignment>
       <master_timeout>
          <sec>3</sec>
          <nanosec>0</nanosec>
```

```
            </master_timeout>
        </replication_settings>
        <service_qos>
            <shared_reader_queue_replication/>
        </service_qos>
    </queuing_service>
```

The following configuration worked:

```
    <queuing_service name="RemoteConfigQS">
        <service_qos>
            <shared_reader_queue_replication>
                <enabled>true</enabled>
                <replication_settings>
                    <queue_instances>3</queue_instances>
                    <synchronize_consumer_assignment>
                        false
                    </synchronize_consumer_assignment>
                    <master_timeout>
                        <sec>3</sec>
                        <nanosec>0</nanosec>
                    </master_timeout>
                </replication_settings>
            </shared_reader_queue_replication>
        </service_qos>
    </queuing_service>
```

This problem has been resolved. Now both of the above configurations will work.

[RTI Issue ID QUEUEING-530]

4.2 Unexpected Error Message when not using <distribution> Tag while Configuring SharedReaderQueue

If you did not use the <distribution> tag under <queue_qos> when configuring a SharedReaderQueue, you may have seen this error message:

```
QUEUEDequeueProcessor_new:Processor DELIVERY_MODE property not set
```

This error message did not affect correctness. This problem has been resolved.

[RTI Issue ID QUEUEING-540]

4.3 Incorrect Publication Rate of Monitoring Data when Setting <status_publication_period> under <monitoring>

In some situations, the <monitoring>/<status_publication_period> tag was not read correctly from the configuration file. This lead to incorrect publication rates of monitoring data. This problem has been resolved.

[RTI Issue ID QUEUEING-545]

4.4 Enabling Monitoring Required Explicitly Setting <monitoring>/<enable>

To enable monitoring, you had to explicitly set the <monitoring>/<enable> tag to true. This behavior has changed. Now, if the <monitoring> tag is present, monitoring is enabled by default, even if <enable> is not explicitly set to true.

[RTI Issue ID QUEUEING-546]

4.5 Invalid Throughput/Latency Statistics when using <statistics>/<statistics_sampling_period> to Configure Sampling Period

When setting the <statistics>/<statistics_sampling_period> tag in the configuration file, the values were not read correctly. This caused incorrect throughput and latency statistics calculations for a SharedReaderQueue. This problem has been resolved.

[RTI Issue ID QUEUEING-547]

4.6 Unexpected Error Message "on file message count (metadata) x != counted standing messages y" when Restoring Persistent SharedReaderQueue

You may have seen the following error message after a persistent SharedReaderQueue with in-memory state was restored after a graceful shutdown.

```
on file message count (metadata) x != counted standing messages y
```

This error message did not affect correctness. This problem has been resolved.

[RTI Issue ID QUEUEING-549]

4.7 Some Undelivered Messages not Resent after Restarting Persistent SharedReaderQueue with In-Memory State

Some undelivered messages may not have been resent to QueueConsumers after restarting a persistent SharedReaderQueue with in-memory state.

This problem occurred when *Queuing Service* sent a message to a QueueConsumer and then *Queuing Service* restarted before receiving a negative acknowledgement or timing out the sample delivery to the QueueConsumer. This problem has been resolved.

[RTI Issue ID QUEUEING-551]

4.8 Unable to Change and Persist Queuing Service Command-Line Parameters when Running as Windows Service

In previous releases, you could not change and persist the default command-line parameters to be used by *Queuing Service* when it runs as a Windows Service.

For example, the <custom parameters> values configured with this command were ignored:

```
sc config rtiqueuingservice binPath= "C:\Program Files (x86)\RTI\RTI
Queuing Service <version>\scripts\rtiqueuingservice.bat -service <custom
parameters>"
```

This problem has been resolved. See the *Queuing Service Getting Started Guide* for details on how to change and persist the command-line parameters, as we have transitioned from using sc to using NSSM.

[RTI Issue ID QUEUEING-560]

4.9 .NET QueueProducer::SendSample() was not Thread Safe

The QueueProducer's **SendSample()** operation (and any of its overloaded versions) may have failed with the following error if it was called concurrently from multiple threads:

```
Exception="DDS.Retcode_BadParameter: Exception of type
'DDS.Retcode_BadParameter' was thrown.
   at DDS.DataWriter.write_w_params_untyped(Object instance_data,
WriteParams_t params)
   at RTI.Connext.Queuing.QueueProducer`1.SendSample(T sample,
WriteParams_t writeParams)
   at RTI.Connext.Queuing.QueueProducer`1.SendSample(WriteSample`1 sample)
```

This problem has been resolved.

[RTI Issue ID QUEUEING-565]

4.10 .NET QueueProducer::WaitForAcknowledgements(SampleIdentity, Duration_t) was not Thread Safe

The QueueProducer's **WaitForAcknowledgements(SampleIdentity, Duration_t)** operation did not behave properly when called concurrently. If two or more threads called this operation for different identities, they all may have been unblocked as soon as one of those identities was acknowledged. This would cause threads waiting for the other identities to return an incorrect value.

This problem has been resolved.

[RTI Issue ID QUEUEING-566]

5 Current Limitations

- The QueueProducer and QueueConsumer wrapper APIs are only supported for the .NET API.

6 Available Documentation

Queuing Service documentation also includes:

- **Getting Started Guide** (RTI_Queuing_Service_GettingStarted.pdf)—Provides installation and startup instructions.
- **User's Manual** (RTI_Queuing_Service_UsersManual.pdf)—Describes how to configure and use *Queuing Service*.